Web Services Workflow for Online Data Visualization and Analysis in Giovanni

Stephen Berrick, Monisha Butler, John Farley, Jeff Hosler, Luther Lighty, Hualan Rui

What is Giovanni?

- **G**ES-DISC **I**nteractive **O**nline **V**isualization **AN**d a**N**alysis **I**nfrastructure
- An underlying infrastructure for a growing family of Web interfaces that allows users to analyze and intercompare gridded data interactively online without having to download any data. Through Giovanni, users are invited to discover and explore our data using sophisticated analyses and visualizations.
- **Simple** Web interface, **multi-sensor** data, **tailored** towards somewhat unique user communities
- Current version is 2 (aka G2) debuted mid 2002
  - Simple Perl CGI, HTML templates, and GrADS for data analysis and image rendering
  - Hierarchical configuration files allow for fairly simple reconfigurations of a particular instance or the create of a new instance
- G2 has been enormously successful

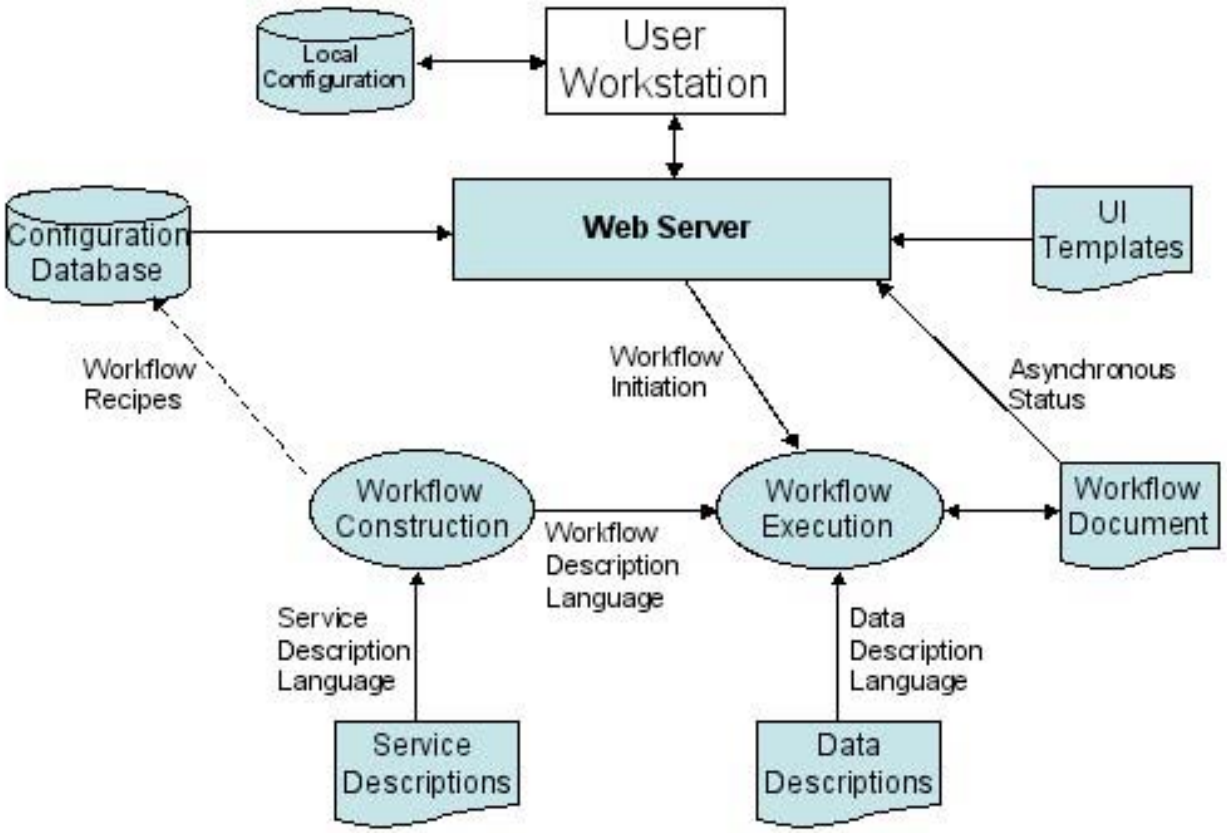
G2 Limitations

- Configuration complexity
  - Not extensible for new requirements
  - Options and defaults set by developers, not users
- Support only for GrADS
  - Difficult to move beyond Level 3 gridded data
  - Rendering options more limited than say, IDL
- Synchronous interface
  - HTTP time out
  - Workarounds: limit resolution, spatial area, or sophistication of analysis
- No well-defined algorithm/renderer interface
  - Practical only for developers to add or modify algorithms and renderers
- Works mostly with local data
  - Can work with GrADS Data Server (GDS)
  - But for performance (HTTP time out), most data are local

G3 Requirements

- Separation of infrastructure from algorithms and image renderers
- Data location transparency
- Agnostic w/r/t algorithm and image rendering language
- Asynchronous
- Workflow paradigm
- Service oriented architecture
- Data lineage
- “My Giovanni”

G3 Architecture Diagram



G3 Architecture (cont.)

- GUI uses CGI, HTML templates, server-side includes (SSI), and styles sheets. A database schema constrains the GUI elements and options available for representing a particular service.
  - Users eventually will be able to customize and save a particular GUI configuration locally as XML document and reconstitute it later.
- Database schema constrains allowable services (processing and image rendering) for each parameter of each data set and which of those data sets, parameters and services are exposed in each Giovanni instance.
- Workflows built from one or more services

Services

- Services described in Service Description Language (SDL), our XML implementation
- Supports command line programs, Web services via WSDL, or Perl/Python functions
- SDL provides:
  - URL to XML schema describing inputs/outputs (XSchema format)
  - Binding describing how to execute service
  - URL to description for GUI and data lineage

Example Service

```
<service>
   <name>mapTimeAverage</name>
<description>http://giovanni.gsfc.nasa.gov/G3/descriptions/mapTime
    Average.shtml</description>

   <serviceType kind = "commandLine">
      <commandLine>
         <executable>mapTimeAverage.pl</executable>
         <schema>mapTimeAverageService.xsd</schema>
         <standardArgs>true</standardArgs>
      </commandLine>
   </serviceType>

</service>
```

Composite Services

- Composite services can be built up from more primitive services
- For example, we have a variety of “averaging” services, some distinguished by the data structures they operate on
- Composite service *Average* can hide the implementation distinctions

Pre-Built Service Categories

- Data Retrieval - Fetch data from source location into temporary cache. Low-level services are available for retrieving data via HTTP, FTP, and OPeNDAP protocols.
- Data Transform - Scaling, subsetting, and regridding.
- Calculation – Average, subtract, add, and generate statistics for the input data.
- Filter – Various filter services are provided to select a subset of the input data (e.g. filter by quality flag)
- Image Rendering –GIF, PNG, GeoTIFF and other formats are provided to output data in ways that are easy to display and print.
- Packaging - For users who want the raw data, instead of images, packaging services are provided to deliver outputs in ASCII, binary, or HDF formats

Recipes

- Recipes are workflows described in simple XML based on Sci-Flo style
- Currently, we have no workflow execution engine (we hope to use Sci-Flo).
- In meantime, we use simple script, genscript, to generate a simple Perl script that executes workflow
- Allows us to concentrate on services rather than on workflow engine development
- Recipes currently constructed manually and integrated into database manually. There will be a GUI eventually (needed for “My Giovanni”)
- Recipes current built offline.

Recipe Example With Three Services

```
<recipe>
   <steps>
      <step>fetcher</step>
      <step>mapTimeStatistics</step>
      <step>areaPlot</step>
   </steps>
</recipe>
```

Recipes (cont.)

- Once constructed, info required to execute the recipe is exported to the Configuration Database, used by the user interface (UI) to dynamically construct an graphical user interface (GUI).
- When user visits the G3 landing page, script is run to access database. User is redirected to a Web page that shows all currently available instances.
- When user clicks on an instance, another script is launched that generates a page showing info about instance and its data, and a Web form that lists all of the available parameters, services, and associated options for that instance.
- Once inputs selected for the workflow from the GUI, the UI creates an XML representation of the inputs & initiates execution of the appropriate workflow.

Workflow State

- When workflow is initiated, a workflow document is created for that specific execution
- Contains inputs/output along with services
- Each service updates this document. Thus, document serves as a state repository
- Via XLS transforms, state is provided to user via GUI or via RSS (asynchronous way for user to get product)

```
<inputs>
<dbID>82732323</dbID>
<event>(45.6,-101.2)</event>
</inputs>
<process>
<name>getDatapoolInfo.pl</name>
<stepNmbr>1</stepNmbr>
<status>COMPLETE</status>
<inputs>82732323</inputs>
<outputs>/datapool/OPS/hdf.dat</outputs>
</process>
<process>
<name>HdfLook.pl</name>
<stepNmbr>2</stepNmbr>
<status>active</status>
<inputs>/datapool/OPS/hdf.dat</inputs>
<outputs></outputs>
</process>
<process>
<name>MapServer.pl</name>
<stepNmbr>3</stepNmbr>
<status>pending</status>
<inputs></inputs>
<outputs></outputs>
</process>
```

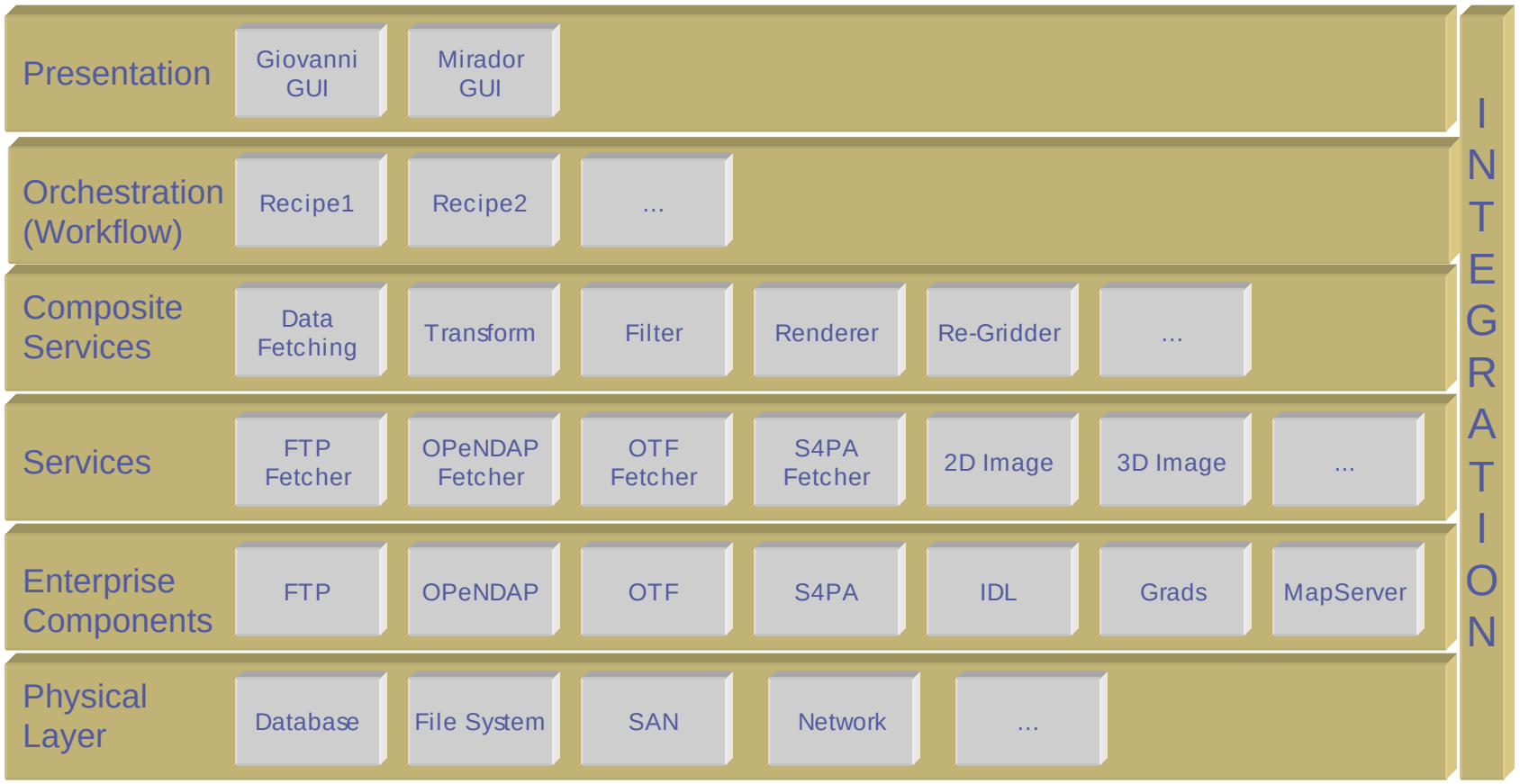
Example of WorkFlow XML

```
my $stat =
G3Xml::setXmlFileName(mapping.xml,step1);
my $dbid = getDBID()
...
my $path = findDatapoolLocation();
writeOutput($path)
```

```
my $stat =
G3Xml::setXmlFileName(mapping.xml,step2);
my $dbid = getDataPoolPath()
my $file = runHDFLook()
writeOutput($file)
```

I/O Example of 2 services

SOA Architecture Stack



G3 User Interface

- Most of remaining work in G3 is in the UI
- Provide users ability to save UI preferences (not just cosmetic)
- Considering using Ajax.
- Must consider usability issues and Section 508.

Still To Do

- GUI tools to manage database
- Workflow execution via Web Service

Acknowledgments

- Giovanni core team lead by Hualan Rui:
  - Timothy Dorman, Dr. Zhong Liu, Dr. Suhung Shen, Xiaoping Zhang, and Tong Zhu
- Scientists lead by Dr. Eyal Amitai:
  - Dr. James Acker, Dr. Suraiya Ahmad, Arun Gopalan, James Johnson, Dr. Gregory Leptoukh, Jason Li, Dr. Jianping Mao, Dr. Andrey Savtchenko, and Dr. William Teng
- Other support:
  - The late Dr. Yoram Kaufman (NASA GSFC)
  - Dr. Watson Gregg (NASA GSFC) through REASoN CAN 02-OES-01

Backup Slides

GrADS

- Grid Analysis and Display System
- http://www.iges.org/grads/
- The Grid Analysis and Display System (GrADS) is an interactive desktop tool that is used for easy access, manipulation, and visualization of earth science data. The format of the data may be either binary, GRIB, NetCDF, or HDF-SDS (Scientific Data Sets). GrADS has been implemented worldwide on a variety of commonly used operating systems and is freely distributed over the Internet.

GrADS Data Server (GDS)

- http://www.iges.org/grads/gds/
- The GrADS Data Server (GDS, formerly known as GrADS-DODS Server) is a stable, secure data server that provides subsetting and analysis services across the internet. The core of the GDS is OPeNDAP (also known as DODS), a software framework used for data networking that makes local data accessible to remote locations. GDS services can be provided for any GrADS-readable dataset.